VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV INTELIGENTNÍCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF INTELLIGENT SYSTEMS

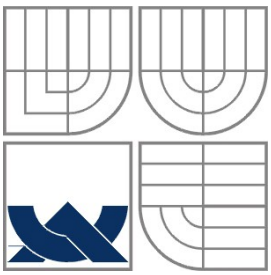
REKONSTRUKCE 3D MODELU OBLIČEJE POMOCÍ ZAŘÍZENÍ KINECT

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

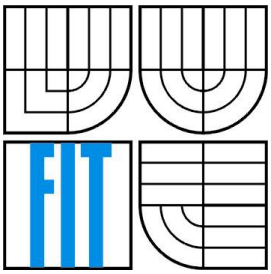
AUTOR PRÁCE
AUTHOR

TOMÁŠ NESVADBA

BRNO 2014



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

REKONSTRUKCE 3D MODELU OBLIČEJE POMOCÍ ZAŘÍZENÍ KINECT

CREATING A 3D FACE MODEL USING KINECT DEVICE

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

Tomáš Nesvadba

VEDOUCÍ PRÁCE
SUPERVISOR

Ing. Štěpán Mráček

BRNO 2014

Abstrakt

Cílem této bakalářské práce je navrhnout způsob pro zvýšení rozlišení 3D modelů obličeje pořízených senzorem Kinect a toto řešení implementovat. V práci jsou popsány metody snímání 3D modelů, senzor Kinect a jeho vlastnosti v porovnání s kvalitními 3D skenery. Hlavní částí práce je popis statistického modelu založeného na Analýze hlavních komponent a jeho využití při návrhu aplikace. V závěru je provedeno testování s různými vstupy a zhodnocení dosažených výsledků implementovaného programu.

Abstract

The main aim of this bachelor's thesis is to propose a method for increasing the resolution of 3D face models captured by Kinect. The document describes different methods of capturing 3D models and compares the Kinect sensor with more acurate scanners. The main part of this document is dedicated to statistic model based on the Principal Components Analysis and its usage in the project. The last section of the paper describes testing of various models and discuss the results and another options of aplication.

Klíčová slova

Java, Java 3D, Analýza hlavních komponent, strukturované světlo, Kinect, tvarovatelný model tváře, OpenCV, rekonstrukce obličeje

Keywords

Java, Java 3D, Principal Components Analysis, structured light, Kinect, morphable face modeler, OpenCV, face reconstruction

Citace

Tomáš Nesvadba: Rekonstrukce 3D modelu obličeje pomocí zařízení Kinect, bakalářská práce, Brno, FIT VUT v Brně, 2014

Rekonstrukce 3D modelu obličeje pomocí zařízení Kinect

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením Ing. Štěpána Mráčka. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

…..……
Tomáš Nesvadba
14. května 2014

Poděkování

Na tomto místě bych rád poděkoval svému vedoucímu Ing. Štěpánovi Mráčkovi za uvedení do dané problematiky a za pomoc a podporu při jejím zpracování.

Obsah

Obsah....1
1 Úvod....3
2 Skenování 3D modelu....4
2.1 Základní principy strukturovaného světla....4
2.1.1 Principy promítání....4
2.1.2 Metody skenování....5
2.1.3 Porovnání metod....5
2.1.4 Transformace obrazu....6
2.2 Skenovací zařízení....7
2.2.1 Konica Minolta Vivid 910 a Vivid 9i....7
2.2.2 NextEngine....7
2.3 Kinect....8
2.3.1 Parametry....9
2.3.2 Možnosti využití....11
2.3.3 Porovnání s ostatními zařízeními....11
3 Rekonstrukce pomocí tvarovatelných modelů obličeje....13
3.1 Tvarovatelné modely....13
3.1.1 Ustanovení korespondence....14
3.1.2 Zarovnání modelů....15
3.1.3 Statistické modelování....15
3.2 Analýza hlavních komponent....15
3.2.1 Výpočet vlastních čísel a vektorů....15
3.2.2 Rekonstrukce pomocí PCA....17
3.2.3 Identifikace pomocí PCA....18
4 Návrh řešení....19
4.1 Rozsah a vlastnosti aplikace....19
4.1.1 WaveFront .obj....19
4.1.2 Databáze....20
4.1.3 Zarovnání obličejů....21
4.1.4 Rekonstrukce....21
5 Implementace....23
5.1 Jazyk a knihovny....23
5.1.1 Java 3D....23
5.1.2 OpenCV....24

5.2 Tvorba databáze....25
5.3 Načtení a zarovnání modelů....25
5.4 Hloubková mapa a interpolace....27
5.5 Výběr plochy pro rekonstrukci....28
5.6 Analýza a rekonstrukce....29
5.6.1 Analýza....29
5.6.2 Rekonstrukce....30
5.6.3 Porovnání modelů....30
6 Testování....31
6.1 Model vyřazený z databáze....31
6.2 Modely obsažené v databázi....32
6.3 Modely z Kinectu....33
6.4 Použití různého počtu vektorů....34
7 Závěr....36
Literatura....37
Příloha A - Obsah CD....39
Příloha B - Manuál....40

1 Úvod

Tato bakalářská práce se zabývá možnostmi rekonstrukce modelů obličeje získaných ze zařízení Kinect od firmy Microsoft Corporation. Kinect je zařízení určené pro snímání 3D modelů za pomoci strukturovaného světla. Jeho primární využití je u herní soustavy Xbox 360 a Xbox One jako konkurent ovládacího zařízení Wii od Nintendo Corporation. Díky využití strukturovaného světla není potřeba fyzického ovladače tak, jak je tomu u Wii. Uživatel může svůj počítač či Xbox ovládat skrze nastavená gesta s využitím celého těla. Vzhledem k cílové skupině, na kterou je senzor zaměřen (široká veřejnost), je jeho cena stanovena tak, aby byl co nejvíce dostupný. Nízká cena se tak projevila na kvalitě zařízení, zejména pak na kvalitě snímaných modelů. Ty jsou dostačující pro ovládání her, nicméně pro získání kvalitnějšího a reálného modelu obličeje jsou nedostatečné.

V nadcházejícím textu budou porovnána různá zařízení využívající strukturovaného světla pro snímání 3D modelů. Dále budou rozebrány možnosti rekonstrukce s využitím tvarovatelného modelu obličeje.

Účelem této práce je navrhnout řešení nedostatečné kvality modelů a toto řešení implementovat a otestovat. Program je vytvořen v jazyce Java. Důraz byl kladen na přehledný program pro běžné použití. Implementačně náročnější problémy budou zmíněny spolu s vhodnými knihovnami pro práci s 3D modely.

V poslední kapitole budou zhodnoceny výsledky rekonstrukce modelů pomocí vytvořeného programu. Testování programu bylo zaměřeno na otestovaní na trénovacích datech a následně na modelech z Kinectu. Na závěr bude nastíněno možné rozšíření programu vzhledem k již implementovaným funkcím.

2 Skenování 3D modelu

Po přečtení této kapitoly získá čtenář povědomí o metodách využívání strukturovaného světla a technologiích založených na těchto principech. Budou probrány možnosti jednotlivých technologií a jejich využití při snímání. Budou popsány kvalitní 3D skenery a podrobněji popsáno zařízení Kinect.

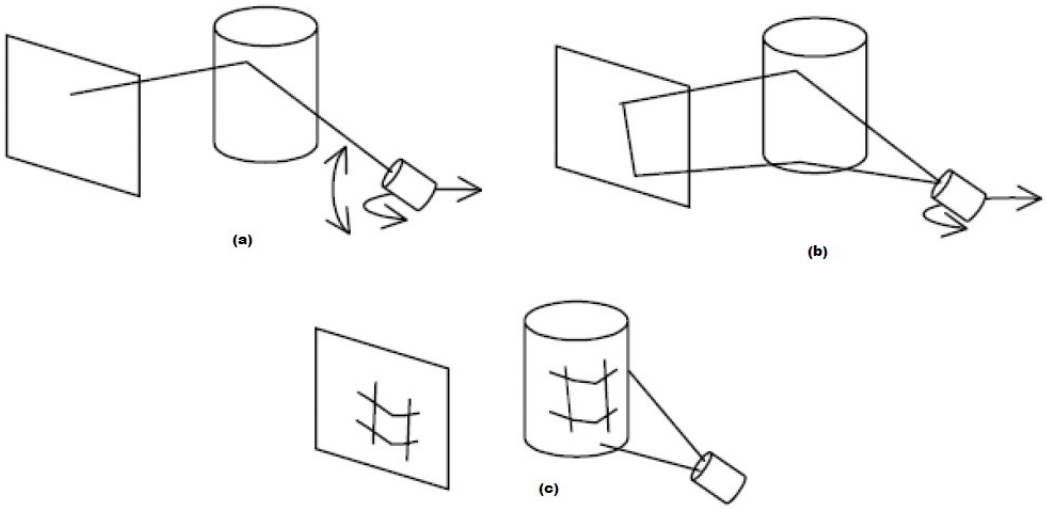
2.1 Základní principy strukturovaného světla

Snímače využívající strukturovaného světla jsou založené na principu promítání světelného vzoru na objekt a jeho zpětné zachycení. Bývají tvořeny jednou a více kamerami a zdroji světla, čímž poskytují kvalitní alternativu stereoskopickým snímačům v kontrolovaném prostředí (průmysl či medicína), ve slabě osvětleném prostředí (noční vidění) nebo ve slabě texturovaném prostředí (biometrie). Nicméně se potýkají s problémem, že vyzařované světlo je invazivního charakteru a může tak narušovat snímané informace o textuře či tvaru [1]. V případě laseru může být až životu nebezpečné. Aby byl plně využit potenciál strukturovaného světla za účelem snímání bez ovlivnění vyzařovaným světlem, byly vytvořeny bezpečné snímače využívající světlo mimo viditelné spektrum.

2.1.1 Principy promítání

Snímač strukturovaného světla je podobný klasickému stereoskopickému senzoru se světelným zdrojem místo kamery. Světelný zdroj může na objekt promítat několik různých vzorů. Těmto vzorům pak musí být skenování přizpůsobeno.

Nejjednodušším vzorem, který je možno na objekt promítat, je jediný bod. To zaručuje přímou korespondenci promítaného a snímaného vzoru, vyžaduje však mechanické skenovaní s promítáním bodu ve vertikální i horizontální ose (viz Obr 2.1a). Pokročilejší formou je promítání skrze štěrbinu, při němž je stále potřeba pohybu projektoru alespoň v jedné ose (viz Obr 2.1b). Korespondence vzorů je pak řešena jedinou ortogonální translací světelného projektoru. Tyto skenovací metody jsou značně zdlouhavé a mechanicky náročné. Řešením je promítání dvoudimenzionálního vzoru (např. mřížky) na objekt (viz Obr 2.1c). Při tomto promítání je nutné zavést kódovaní signálu, aby byly odpovídající si body přesně přiřazeny a bylo dosaženo vyšší kvality [1]. Touto problematikou se zabývali P. M. Will a P. S. Pennington ve své práci, v níž navrhují kódování a dekódování signálu pomocí Fourierovy transformační techniky [2]. Pro další zvyšování kvality se skenování provádí z různých úhlů.



Obr 2.1: Různé druhy promítání (a) jediný bod, (b) skrze štěrbinu, (c) složitější vzory; převzato z [1]

Dalším krokem je transformace získaného obrazu do soustavy souřadnic, kde vznikne digitální 3D model.

2.1.2 Metody skenování

Existuje několik metod využívajících strukturovaného světla o vlnové délce, která je mimo viditelné spektrum, tedy světlo mimo vlnovou délku 380–750 nm. Například metoda infračerveného strukturovaného světla (IRSL – InfraRed Structured Light) využívá blízkého infračerveného spektra (640–2 500 nm) nebo spektra v rozsahu 4 000–15 000 nm. Při použití vlnové délky v rozsahu 300– 1 100 nm není dokonce potřeba infračervené kamery, neboť pro tuto vlnovou délku postačuje CCD kamera. Podobná metoda filtrovaného strukturovaného světla (FSL – Filtered Structured Light) využívá světelný zdroj s přidaným IR filtrem schopným zamezit průchodu světla o vlnové délce nižší než 750 nm, 800 nm atd. Úplně odlišným způsobem je řešena metoda nepostřehnutelného strukturovaného světla (ISL – Imperceptible Structured Light) využívající dvou kamer a speciálního světelného zdroje. Principem je promítnutí vzoru, za nímž následuje jeho komplement o vysoké frekvenci. Jedna z kamer zaznamenává vzor stejně jako v předchozích metodách, zatímco druhá zaznamenává scénu pod uniformním světlem [1]. Výsledkem je 3D model s texturami.

2.1.3 Porovnání metod

Dle závěrů v práci Yvona V. Tadeuze a S. A. Davida F. [1] je nejjednodušeji implementovatelnou metodou skenování pomocí IRSL. Tato metoda nabízí vysokou přesnost a rozlišení, ale trvá dlouho

díky mechanickému skenování a neumožňuje kódování. Její velkou výhodou je možnost venkovního použití, kde je díky robustnosti systému odolná vůči ambientnímu světlu.

ISL je sice pro venkovní použití nevhodný z důvodu dlouhé kalibrace, jeho možnosti jsou ale velké. Díky dvěma kamerám je velmi adaptivní.

FLS oproti IRSL zvládá kódování signálu a má taktéž vysoké rozlišení. FSL ale nedosahuje kvality IRSL při použití laseru nebo kvalit ISL při použití video-projektoru. FSL nabízí kompromis mezi IRSL a ISL. Další vybrané rozdíly jsou popsány v tabulce níže (Tabulka 1).

Tabulka 1: Porovnání metod skenování

	IRSL	ISL	FSL
Laser	Ano	Ne	Ano
Video-projektor	Náročné	Ano	Ano
Kódování barev	Ne	Ano	Ne
Binární kódování	Náročné	Ano	Ano
Hybridní kódování (neviditelné a viditelné)	Ne	Ano	Ano
Mechanické skenování	Ano	Ne	Ne
Barva/Textura/Okraje	Ano	Ano	Ano
Analýza pohybu	Náročné	Ano	Náročné
Venkovní použití	Ano	Náročné	Ano

2.1.4 Transformace obrazu

Získaný obraz strukturovaného světla je nutné transformovat do souřadného systému v několika krocích v závislosti na použitém vzoru promítnutého na objekt a kameře, která obraz zaznamenala. Nejvhodnějším matematickým modelem pro tento úkol se ukázal Pinhole Camera Model, skládající se ze série pěti transformací [3].

1. Transformace reálné souřadnice → kamerové souřadnice
2. Projekce perspektivy
3. Přepočet zakřivení čočky kamery
4. Transformace kamerové souřadnice → pixelové souřadnice
5. Inverzní transformace pixelové souřadnice → reálné souřadnice

2.2 Skenovací zařízení

V této části budou popsána dvě skenovací zařízení od různých výrobců tak, aby čtenář získal představu o současných 3D skenerech a mohl tak porovnat kvalitu skenování u Kinectu.

2.2.1 Konica Minolta Vivid 910 a Vivid 9i

Konica Minolta Vivid 9i (viz Obr 2.2) vyvinutá 3DScanCo je jedním z nejvyspělejších laserových 3D skenerů. Vivid 9i je kombinován PSC-1 fotogrammetrickým[1] systémem. Výsledkem jsou velmi přesně zachycená data z povrchu. Uváděná přesnost skeneru je až 50 μm. Toho je však možné docílit pouze u nepohyblivých předmětů a nemusí to být vhodné pro skenování větších předmětů. Levnější variantou vhodnou pro skenování tváří nebo soch je Vivid 910. Ten je schopen naskenovat 307 000 bodů za 2.5 sekundy nebo 77 000 bodů za 0.3 sekundy ve zrychleném módu. Jeho přesnost je přibližně 4–8krát nižší než u verze 9i. Jeho rozlišení dosahuje 220–400 μm v závislosti na zvoleném módu skenování. Dle firmy 3DScanCo je Vivid 910 vhodný pro skenování architektury, tváří nebo soch [4].

Obr 2.2: 3D skener Vivid 9i; převzato z [4]

2.2.2 NextEngine

Dalším kvalitním skenerem založeným na laserovém snímání je 3D Scanner HD (viz Obr 2.3) od firmy NextEngine, který má přesnost až 100 μm. Přístroj je dodáván spolu s otočným systémem

1 Fotogrammetrie se zabývá rekonstrukcí tvarů, měřením rozměrů a určováním polohy předmětů z fotografických snímků.

AutoDrive a softwarem pro skenování. Možností je i instalace ovladače do programu SolidWorks. Použití tohoto laserového skeneru pro zachycení obličeje je zcela bezpečné, protože světelný paprsek dosahuje pouze jedné tisíciny laserového ukazovátka. Nicméně člověk by se měl vyvarovat přímého pohledu do paprsku.

3D Scanner HD je schopný zachytit texturu v rozlišení 150 nebo 400 DPI v závislosti na používaném módu. Na tom je závislé i zachycení 3D modelu s přesností buď 0.127 mm nebo 0.381 mm. Skenování probíhá průměrně rychlostí 50 000 bodů za sekundu, přičemž typické skenování trvá 2 minuty [5].

Obr 2.3: 3D Scanner HD a AutoDrive; převzato z [5]

2.3 Kinect

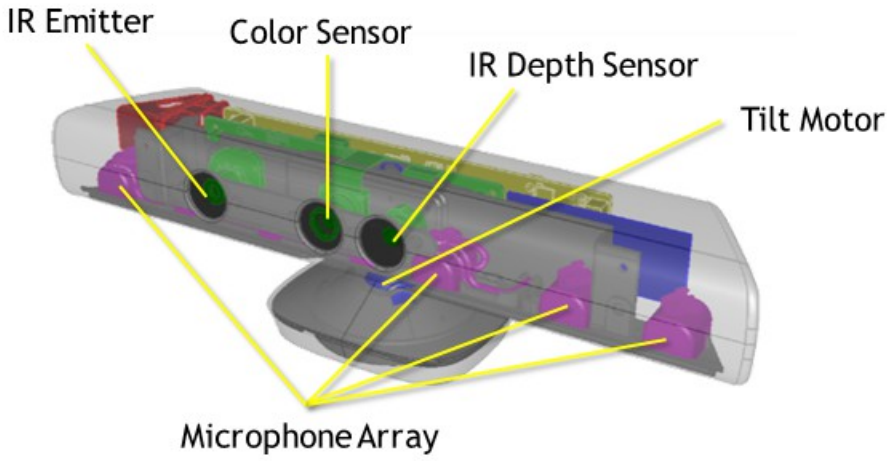
Kinect Sensor (viz Obr 2.4) je snímací zařízení vyvinuté firmou Microsoft Corporation v roce 2010. Kinect první generace byl určen pro herní konzoli Xbox 360. Patří mezi nejrychleji prodávanou spotřební elektroniku. Kinect umožňuje přirozeným pohybem celého těla ovládat konzoli Xbox či počítač. To je možné díky dlouhodobému výzkumu na poli počítačového vidění [6].

Příznivá cena Kinectu umožnila i další využití mimo herní průmysl (např. robotika, 3D modelování, telekonference). Orientace pouze na platformu Windows PC a Xbox 360 umožňovala využití snímaných dat pouze pomocí dostupných oficiálních nástrojů. Na jiných platformách je možné využít open source knihovny Libfreenect nebo OpenNI+SensorKinect umožnujících práci s využitím různých programovacích jazyků (Python, Java, atd.). V roce 2012 byl Microsoftem uvolněn oficiální nástroj Kinect SDK pro programovací jazyky C++, C# a Visual Basic [7].

Obr 2.4: Senzor Kinect; převzato z [6]

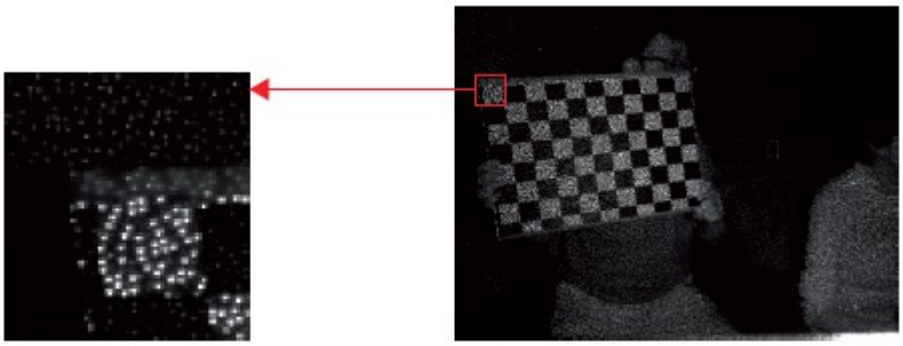
2.3.1 Parametry

Kinect Sensor se skládá z infračerveného projektoru kombinovaného s infračervenou kamerou. Metoda získávání hloubky obrazu – tzv. „depth sensing technology“, se zakládá na technologii od izraelské firmy PrimeSense Ltd., která byla založena roku 2005 a zabývá se strojovým 3D viděním pro digitální zařízení. Jejím hlavním produktem je PrimeSensor a k němu patřící vývojová knihovna OpenNI [8]. Jedná se o systém na čipu, který dokáže vidět, sledovat a reagovat na pohyby uživatele. Své uplatnění našel ve videohrách, konzumní elektronice, domácích trenérech a videokomunikacích. Můžeme ho najít i u konkurenčního zařízení od ASUSu zvaného WAVI Xtion, jehož prodej byl spuštěn v roce 2011 [9]. Od roku 2013 PrimeSense Ltd. funguje jako poddivize Apple Inc. [10].



Obr 2.5: Vnitřní stavba senzoru Kinect; převzato z [12]

Způsob promítání je řešen dle principů strukturovaného světla popsaného výše. V tomto případě projektor na objekt promítá specifický vzor bodů (viz Obr 2.6), který je patentovaný firmou PrimeSense [11]. Rekonstrukce je pak řešena pomocí triangulace [6].



Obr 2.6: Promítaný vzor bodů; převzato z [6]

Záznam hloubky je prováděn monochromním CMOS IR senzorem, který dokáže rozlišovat ve vertikální a horizontální ose s milimetrovou přesností a hloubku s přesností přibližně 1 cm. Výstupní obrazy je možné získávat s frekvencí 30 Hz o rozlišení VGA (640×480 pixelů) [6]. O stejném frekvenci je zaznamenáván RGB obraz druhým CMOS senzorem o rozlišení 1280×960. Zorný úhel snímače je 43 stupňů vertikálně a 57 stupňů horizontálně. Díky vestavěnému motorku v základně je možné, aby se senzor otáčel ve vertikále o 27 stupňů nahoru či dolů [12].

Na Kinectu jsou celkem 4 mikrofony připevněné na spodní hraně snímače (viz Obr 2.5). Mikrofony obsahují 24-bitový ADC převodník a jsou schopné potlačit akustické echo či šum. Díky svému rozmístění jsou schopné rozpoznat směr, ze kterého přichází zvuk a dokonce i původce zvuku [12].

2.3.2 Možnosti využití

Kinect díky své dostupnosti umožnil vznik mnoha dalších využití, která nemusela být při vývoji zařízení evidentní. Podle počtu webových stránek, které neustále přibývají, je zřejmé, že zájem o zařízení je značný.

> ‘‘Every few hours new applications are emerging for the Kinect and creating new phenomenon that is nothing short of revolutionary.’’ KinectHacks.com

Nepočítáme-li hry, pak můžeme využití najít například v robotice, kde Kinect slouží pro rozpoznávání okolí a umožňuje tak robotovi pohybovat se v prostředí tak, aby se vyhýbal překážkám. Příkladem je auto, které si vybírá cestu pomocí reaktivního navigačního přístupu zvaného „tentacles“ nebo létající robot schopný následovat vytyčené body v prostoru [13].

S využitím více senzorů je také možné pořádat příjemnější videokonference. Díky využití více senzorů lze udržovat oční kontakt s lidmi zapojenými do konference. Je tak vytvořena reálnější a

přirozenější komunikace. Při vhodném nastavení je možné dosáhnout i soukromého rozhovoru v rámci konference pouhým nakloněním se k druhé osobě.

2.3.3 Porovnání s ostatními zařízeními

Kinect zdaleka nedosahuje kvality snímání nejmodernějších senzorů jako například Vivid 910 či 3D Scanner HD. Kvalita snímání není však u Kinectu hlavním cílem. Jde o rychlost zpracování dat, aby měl uživatel pocit přirozenosti ovládání a v tomto směru s frekvencí 30 snímků za sekundu předstihuje výše zmíněné zástupce. Podstatný rozdíl je také v ceně – cena Kinectu a většiny ostatních skenerů se liší až více než 20krát.

Tabulka 2: Porovnání skenovacích zařízení

	Kinect	Minolta Vivid 910	3D Scanner HD
Výrobce	Microsoft	3DScanCo	NextEngine
Cena[2]	3 199 Kč	198 225 Kč	59 374 Kč
Parametry			
Mód skenování	standardní	normální a zrychlený	normální a zrychlený
Rozlišení	1 mm	0.220 nebo 0.400 mm	0.127 nebo 0.381 mm
Rozlišení hloubky	100 mm	0.220 nebo 0.400 mm	0.127 nebo 0.381 mm
Rychlost snímání	0.016 s	0.3 s nebo 2.5 s	120 s
Použití			
Software	Kinect SDK	Geomagic caprure	Scanstudio HD Solid Works
Vhodné využití	Xbox hry ovládání počítače	postavy architektury umělecká díla	menší objekty

2 Při kurzu USD 1$ =19.82 Kč

3 Rekonstrukce pomocí tvarovatelných modelů obličeje

V této kapitole budou rozebrány možnosti tvarovatelného modelu obličeje a jejich využití pro rekonstrukci a rozpoznání modelů. Podrobně bude popsáná statistická metoda – analýza hlavních komponent (PCA – Principal Components Analysis) pro vytvoření parametrického tvarovatelného 3D modelu obličeje.

3.1 Tvarovatelné modely

Pro účely rozpoznávání a rekonstrukce 3D modelů se použití statistického modelu 3D povrchu obličeje ukázalo jako velmi slibným směrem pro další zkoumání. Hlavní výhodou použití tvarovatelných modelů je minimalizace manuální činnosti člověka při tvorbě modelu. Další možné využití může být při modelování obličeje při aplikaci dalších parametrů, kterými mohou být například váha, věk, pohlaví, výraz atd. Nejvýznamnější práci na tomto poli provedl Blanz a Vetter [14].

Nejdůležitějším cílem při modelování obličejů je snaha tvořit reálné obličeje a vyjmout z procesu obličeje nereálné. To přináší několik problémů, které je nutné řešit. Při tvorbě tvarovatelných modelů je důležité zarovnání povrchů. Existuje velké množství technik, které zarovnají povrchy na sebe. Většina z nich zahrnuje vyznačení charakteristických rysů na modelu. Nejtypičtějšími rysy pro zarovnávání jsou oči, nos, ústa a okraj obličeje, protože jsou snadno měřitelné. Mnohem hůře se dá změřit, zda se jedná o mužskou či ženskou tvář nebo zda se osoba usmívá či mračí.

Vetter a Blanz ve své práci [14] pracovali s anotovanými modely. U každého modelu byly poznačeny atributy jako pohlaví, výraz, rasa, plnost tváří apod. Všechny použité obličeje měli k dispozici s různými výrazy. Obličeje byly nejdříve převedeny na sloupcový vektor a následně pomocí výpočtu (1) získali vektor, jehož přidáním do databáze mohli měnit výraz tvarovatelného modelu. Ve vzorcích (1,2,3) S zastupuje vektor tvaru obličeje (souřadnice) a vektor T texturu (barvu).

$$\begin{aligned} \Delta S &= S_{výraz} - S_{normální} \\ \Delta T &= T_{výraz} - T_{normální} \end{aligned} \tag{1}$$

Na rozdíl od tvářových výrazů, invariantní atributy obličeje jsou těžší na izolování. Pomocí následujícího výpočtu je možné modelovat rysy jako pohlaví, konkávnost nosu či plnost tváří.

$$\Delta S = \sum_{i=1}^{m} \mu_i (S_i - \bar{S})$$
$$\Delta T = \sum_{i=1}^{m} \mu_i (T_i - \bar{T}) \quad (2)$$

Modelování obličeje na základě vypočtených atributů je založeno na předpokladu, že funkce μ(S,T) popisující znaky/tvar obličeje je lineární funkcí. Pak platí, že odečtením či přičtením znaku ΔS docílíme změny výrazu.

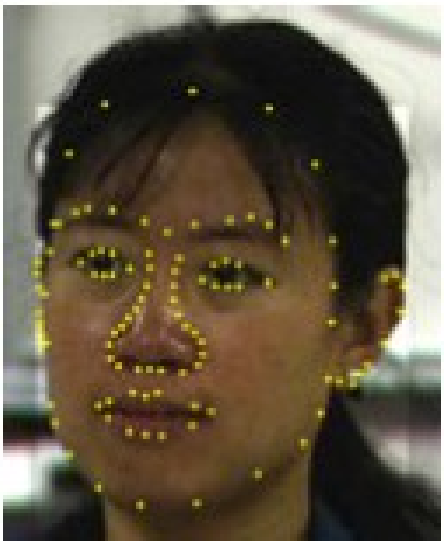
$$\mu_{výraz} = \mu_{normal}(S + \bar{S}, T + \bar{T}) \quad (3)$$

Obecně pro vytvoření statistického modelu platí následují postup:

1. Ustanovení korespondence
2. Zarovnání povrchu modelů
3. Statistické modelování

3.1.1 Ustanovení korespondence

Jedná se o určení všech bodů, podle kterých se bude model zarovnávat na ostatní. Toho je možné docílit manuálně u menších databází nebo automaticky pomocí algoritmizace. Čím přesněji jsou body vybrány, tím kvalitnější může vzniknout databáze (viz Obr 3.1). Typickými body pro zarovnání jsou oči, obočí, ústa, okraj tváře, uši a nos. Pro zarovnání je možné využít i ploch.



Obr 3.1: Body pro zarovnání modelu; převzato z [27]

3.1.2 Zarovnání modelů

Nejdůležitější krok předzpracování obrazu před statistickým zpracováním je zarovnání modelů. Modely zarovnáváme tak, abychom dosáhli invariantnosti u atributů velikosti či náklonu hlavy. Zarovnání se provádí manuálně, dle vybraných bodů (viz 3.1.1), nebo automaticky.

Pro automatické zarovnání lze použít například iterativní zarovnávání popsané v práci Franka B. Haara a Remco C. Veltkampa [15].

3.1.3 Statistické modelování

Při statistickém modelování dochází k vytváření pravděpodobnostního modelu, který popisuje, jak je dvě a více náhodných proměnných korelováno. Pro toto modelování se nejčastěji používá analýza hlavních komponent.

3.2 Analýza hlavních komponent

Analýza hlavních komponent (dále jen PCA) je statistická metoda, která našla uplatnění při rozpoznávání tváří, kompresi obrázku či hledání vzorů v datech o velkých dimenzích. PCA umí predikci, odstranění redundantních dat, extrakci příznaků, kompresi dat a další [16]. Výsledkem analýzy je vektor vlastních čísel (udávajících významnost) seřazených sestupně a korespondující matice vlastních vektorů (udávajících varianci a korelaci). Nejvyšší vlastní čísla značí vektory s nejvyšší variancí. Tyto vektory bývají pak použity pro rekonstrukci.

3.2.1 Výpočet vlastních čísel a vektorů

Výhodou analýzy hlavních komponent je redukce dimenzí. Díky tomu dokážeme pomocí malého množství vlastních vektorů rekonstruovat velké množství obrazů. Cílem této analýzy je získat seřazený vektor vlastních čísel a korespondující matici vlastních vektorů. Vlastní čísla udávají, jakou variabilitu vlastní vektor nese. Čím vyšší hodnota, tím významnější vektor. Všechny použité vzorce v této kapitole vycházejí z prací [16], [17] a [18].

Mějme databázi obrazů nebo databázi různých měření s více parametry. Pro analyzování těchto dat použijeme následujícího postupu.

Hodnoty měření nebo jednotlivé pixely obrazu poskládáme do sloupcového vektoru, kde hodnoty x_i značí libovolný parametr měření.

$$\vec{x} = [x_1, x_2, x_3, x_4, x_5, x_6, \ldots, x_n]^T \tag{4}$$

Tyto vektory poskládáme do matice X obsahující všechna měření, kde první index značí parametr měření a druhý číslo měření či číslo trénovacího obrazu.

$$X = \begin{vmatrix} x_{1,1} & x_{1,2} \ldots & x_{(1,p)} \\ x_{2,1} & x_{2,2} \ldots & x_{(2,p)} \\ x_{3,1} & x_{3,2} \cdots & x_{(3,p)} \\ \ldots & \ldots \cdots & \ldots \\ x_{(n,1)} & \ldots \cdots & x_{(n,p)} \end{vmatrix} \tag{5}$$

V následujícím kroku vypočítáme průměrnou hodnotu *m* každého parametru a následně sestavíme vektor *M* značící průměrnou hodnotu všech parametrů *x*.

$$\begin{aligned} m &= \frac{1}{P} \sum_{i=1}^{n} X_i \\ M &= [m_1, m_2, m_3, m_4, \ldots m_n]^T \end{aligned} \tag{6}$$

Data matice *X* vycentrujeme odečtením průměru v každém řádku. Získáme tak matici, kde součet hodnot ve sloupci je roven nule a konkrétní hodnoty jsou odchylkami od průměrného vektoru.

$$\overline{x_i} = \vec{x}_i - M \tag{7}$$

Z takto vypočtené matice vypočteme kovarianční matici, kterou získáme ze vztahu (8) nebo vztahu (9).

$$C = \overline{X} * \overline{X^T} \tag{8}$$

$$\acute{C} = \overline{X^T} * \overline{X} \tag{9}$$

Podstatným rozdílem mezi oběma vztahy je paměťová náročnost. V prvním případě získáme matici o velikosti N×N. Jednoduchým propočtem zjistíme, že vektor obrazu o rozlišení 100×100 pixelů by měl 10 000 parametrů a následná kovarianční matice by měla dokonce 100 000 000 prvků. V případě rozlišení 400×400 dokonce 25 600 000 000 prvků. V přepočtu do počítačové paměti by takový objekt měl při datovém typu o velikosti 1 B přibližně 23,8 GB. Proto platí, že v případě, že $P \ll N$, je vhodné použít vztah (9) pro výpočet kovarianční matice, neboť velikost kovarianční matice je pouze P×P.

Pro kovarianční matici *C* vyřešením rovnice, získáme vektor λ vlastních čísel a vlastní vektory v matici *V*.

$$C * V = \lambda * V \tag{10}$$

Z důvodů popsaných výše je vhodné pracovat s kovarianční maticí *C´*. Následující úpravou vzorce (10) zajistíme, že výpočet bude paměťově nenáročný a získáme žádané hodnoty.

$$C * V = \lambda * V$$

$$C' * V = (\overline{X^T} * \overline{X}) * V = \lambda * V$$

$$\overline{X} * (\overline{X^T} * \overline{X}) * V = \overline{X} * \lambda * V = \lambda * (\overline{X} * V) \tag{11}$$

$$C * (\overline{X} * V) = \lambda * (\overline{X} * V)$$

Ze vztahu (11) jsme vypočetli matici vlastních vektorů, které však odpovídají kovarianční matici C'. Hodnoty vlastních čísel zůstávají stejné, a tak je možné získat „pravé vlastní vektory“ ze vztahu (12).

$$\acute{V} = \overline{X} * V \tag{12}$$

Následuje normalizace na jednotkovou velikost pro každý vektor v_i.

$$V_i = \frac{v_i}{\sqrt{\sum v_i^2}} \tag{13}$$

V současné chvíli máme vypočteny a normalizovány vlastní vektory uspořádané v matici V, popřípadě V', a vektor vlastních čísel, kde jednotlivé hodnoty korespondují s vlastními vektory. Nyní je nutné seřadit vlastní hodnoty v sestupném pořadí. Současně s řazením vlastních čísel probíhá asociativně řazení vlastních vektorů. Po seřazení vektorů je databáze připravená pro rekonstrukci či rozpoznávání.

3.2.2 Rekonstrukce pomocí PCA

Rekonstrukce založená na analýze hlavních komponent vyžaduje natrénovanou množinu dat v podobě seřazených vlastních čísel a vlastních vektorů kovarianční matice C a průměrného vektoru M. Za předpokladu, že je obrázek zarovnaný na databázi, jej opět převedeme na vektor x.

$$x = [x_1, x_2, x_3, x_4, x_5, x_6, \ldots, x_n]^T \tag{14}$$

Základní myšlenka rekonstrukce je popsána rovnicí:

$$x + e = m + \sum_{i=1}^{N} y_i * v_i \tag{15}$$

Kde x značí vstupní vektor o n pixelech, e značí aproximační chybu, v_i vlastní vektor. Y je koeficient lineární kombinace, který získáme ze vztahu (16).

$$y = V^T * (x - m) \tag{16}$$

3.2.3 Identifikace pomocí PCA

Pro identifikaci osob pomocí analýzy hlavních komponent je nutné databázi obličejů promítnout do vlastních vektorů. Toho dosáhneme následujícím vztahem:

$$\tilde{x}_i = \bar{x}_i * V^T \tag{17}$$

Vstupní obraz pro identifikaci převedeme na vektor *u* a vycentrujeme.

$$\bar{u} = u - M \tag{18}$$

Následně vypočítáme vlastní vektor.

$$\tilde{u} = \bar{u} * V^T \tag{19}$$

Hledání nejpodobnější tváře probíhá porovnáváním obrazů z databáze. Využít k tomu můžeme například Euklidovu (20) nebo Hammingovu (21) metriku, kde hledáme minimální hodnotu *d* [17].

$$d_E(\tilde{u}, \tilde{x}) = \sqrt{\sum_{j=1}^{P} (\tilde{u}_j - \tilde{x}_j)^2} \tag{20}$$

$$d_H(\tilde{u}, \tilde{x}) = \sqrt[n]{\sum_{j=1}^{P} |(\tilde{u}_j - \tilde{x}_j)|} \tag{21}$$

4 Návrh řešení

Součástí práce bylo vytvoření aplikace, která by byla schopna vylepšit obrazy získané pomocí zařízení Kinect. Hlavní myšlenkou aplikace je využití tvarovatelného modelu obličeje, který získáme pomocí analýzy hlavních komponent. Nejdůležitějším rozhodnutím během návrhu bylo, zda bude proces skenování obličeje zahrnutý v programu. Vzhledem k dostupnosti programů pro skenování (ReconstructMe [19], Skanect [20]) bylo od skenovacího procesu upuštěno a program se tak soustředí na zlepšení kvality vstupního souboru. Toto rozhodnutí bylo vykompenzováno volbou vstupních a výstupních souborů ve formátu Wavefront .obj, které jsou běžným formátem pro ukládání 3D modelů.

4.1 Rozsah a vlastnosti aplikace

Při zhotovování návrhu aplikace byly požadované funkce pro vytvoření aplikace zřejmé. Bylo nutné zhotovit tvarovatelný model obličeje, načíst model z Kinectu a model rekonstruovat. Tyto cíle můžeme dále rozdělit na podproblémy, které se musí při implementaci řešit.

Konstrukce statistického modelu:

- Výběr vhodné databáze obličejů a selekce obličejů
- Načtení databáze
- Určení korespondence a zarovnání modelů na sebe
- Převod modelů do hloubkové mapy
- Statistická analýza

Model z Kinectu:

- Naskenování obličejů a uložení ve formátu .obj
- Načtení modelu
- Určení korespondence a zarovnání modelu na obličeje v databázi
- Provedení rekonstrukce
- Uložení výsledku

4.1.1 WaveFront .obj

Soubory .obj jsou vyvinuté firmou Wavefront Technologies pro jejich animační grafický balíček Advanced Visualizer. Jedná se o soubory popisující geometrii objektů. Dnes se jedná o rozšířený

formát, který je kompatibilní s téměř všemi programy pro 3D modelování [21]. Pro prohlížení těchto souborů můžeme využít například programy MeshLab, quick3D 4.0 Geometry, Browz 3D 1.0 nebo GLC_Player. Wavefront .obj je textový soubor popisující jmenovitě pozici každého vertexu, normály, plochy a textury. Je čitelný i pro běžného uživatele a je tak možné menší objekty (krychle, hranol, apod.) vytvořit manuálně. Je nutné však dodržet strukturu souboru (viz Obr 4.1).

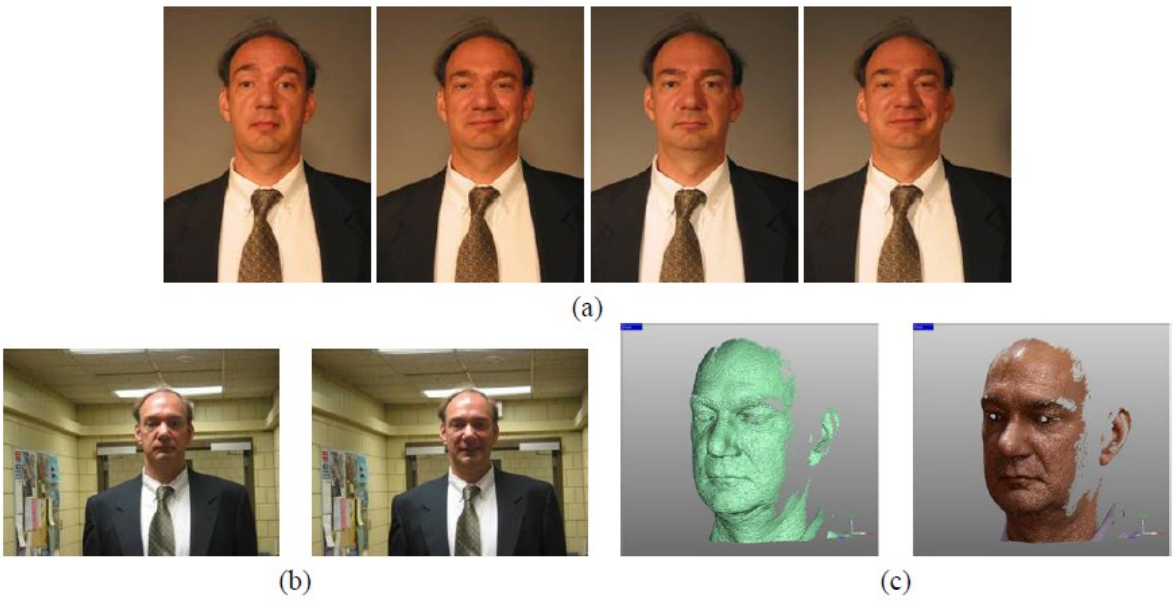
```
# List of Vertices, with (x,y,z[,w]) coordinates, w is optional and defaults to 1.0.
v 0.123 0.234 0.345 1.0
v ...
...
# Texture coordinates, in (u ,v [,w]) coordinates, these will vary between 0 and 1, w is optional and default to 0.
vt 0.500 1 [0]
vt ...
...
# Normals in (x,y,z) form; normals might not be unit.

vn 0.707 0.000 0.707
vn ...
...
# Parameter space vertices in ( u [,v] [,w] ) form; free form geometry statement ( see below )
vp 0.310000 3.210000 2.100000
vp ...
...
# Face Definitions (see below)
f 1 2 3
f 3/1 4/2 5/3
f 6/4/1 3/5/3 7/6/5
f ...
```

Obr 4.1: Vnitřní stavba souboru .obj

4.1.2 Databáze

Pro tvorbu databáze byla zvolena data pořízená Univerzitou Notre Dame, která v rámci projektu Face Recognition Grand Challenge (dále jen FRGC), vytvořila databázi čítající 50 000 záznamů. Projekt vznikl z důvodu velkého zájmu vývojářů o nové techniky rozpoznávání a rekonstrukce obličejů. Vývojářům je tak umožněno jednotného srovnání vyvíjených metod. Databáze obsahuje 3D i 2D snímky o vysokém rozlišení skenované za různých podmínek (viz Obr 4.2). Skenování 3D snímků probíhalo v kontrolovaném prostředí vhodného pro senzor Minolta Vivid 900/910. Osoby byly skenovány dvakrát. Jednou s výrazem (úsměv) a jednou bez výrazu [22].

Pro tvorbu statistického modelu jsem se rozhodl, že z této databáze vyberu vždy jen jeden záznam neobsahující výraz za každou osobu. Dále pak tyto jednotlivce rozdělím podle pohlaví. Vzhledem k faktu, že modely nejsou anotované, třídění probíhalo manuálním výběrem, kde mohlo dojít k subjektivnímu hodnocení tváře a ke špatnému zařazení. Výsledkem této selekce byl výběr 72 jednotlivců v zastoupení 40 mužů a 32 žen.



Obr 4.2: Vzorový záznam v databázi FRGC (a) Snímky při kontrolovaném osvětlení (b) Snímky při nekontrolovaném osvětlení (c) 3D model včetně textury; převzato z [28]

4.1.3 Zarovnání obličejů

Pro zarovnání obličejů jsem zvolil 5 kontrolních bodů, pomocí kterých zarovnám obličeje na sebe. Těmi body jsou oči, bod nad nosem, bod pod nosem a brada, které jsem zvolil z důvodu zarovnání do os. Oči poskytují vhodnou rovinu pro zarovnání do stejné výšky a hloubky ve scéně. To samé platí i pro nos, kde zarovnám bod pod nosem a nad nosem do stejné hloubky otočením modelu kolem horizontální osy. Bod brady byl využit jako pomocný bod. Ze všech 5 bodů se vypočítá průměrná hloubka a následně se model zarovná na předem definovanou hloubku.

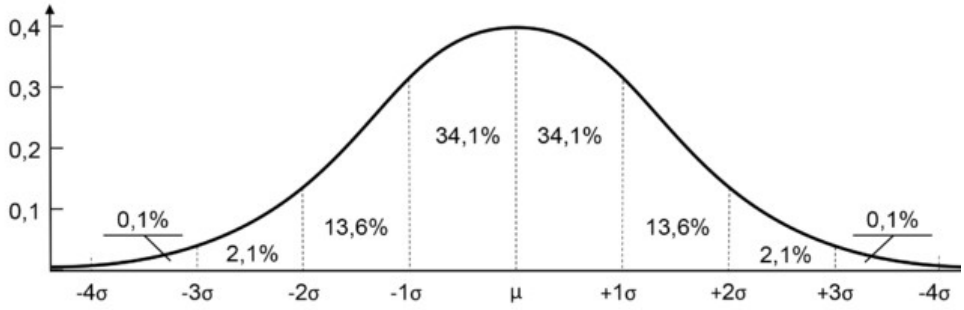
4.1.4 Rekonstrukce

Základní myšlenkou rekonstrukce je promítnutí méně kvalitního modelu obličeje do pravděpodobnostního modelu vytvořeného z kvalitních 3D snímků databáze FRGC. Tím by došlo k redukci dimenzí. Model rekonstruované tváře by se tak přizpůsobil hodnotám a závislostem dle pravděpodobnostního modelu.

Po promítnutí vznikne vektor o délce rovnající se počtu vlastních vektorů databáze s hodnotami vah využití vektorů. Pro zlepšení rekonstrukce bude přidáno ořezání získaných vah w_i pomocí vlastních čísel λ. Hodnoty budou ořezány pomocí vztahu (22), popsaného v práci Taylora a Cootes [23].

$$|w_i| \leq 3 * \sqrt{\lambda_i} \tag{22}$$

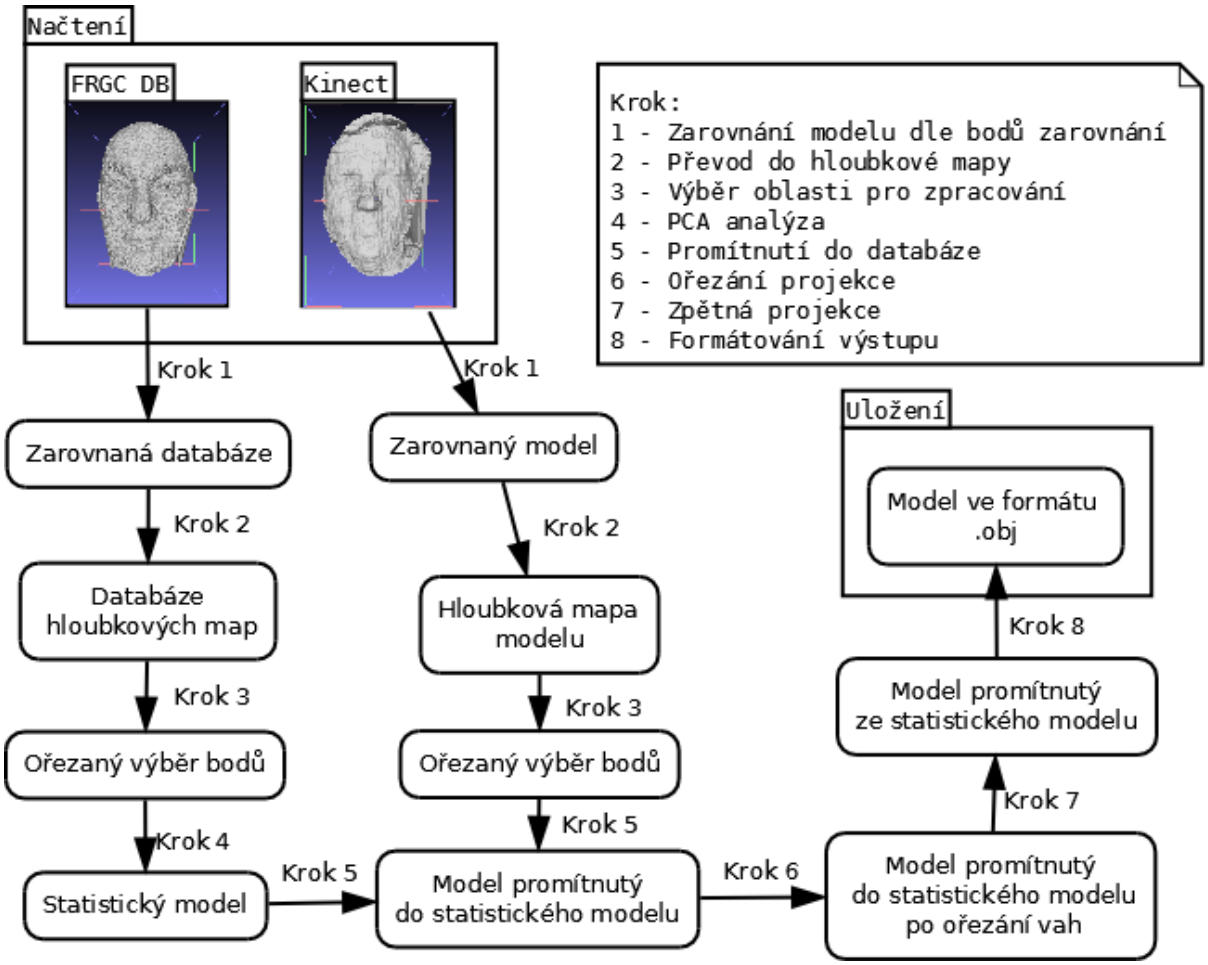
Ořezání vychází z pravidla tří sigma, které předpokládá, že výskyt bodu je gaussovského rozložení (viz Obr 4.3). Pak platí, že 99,73 % všech bodů v Gaussově rozložení je ve vzdálenosti 3 σ od středu μ.



Obr 4.3: Gaussovo rozložení

Po úpravě vah promítneme model zpět a získáme tak kvalitnější model s ostřejšími rysy. Model bude výrazně závislý na zvolené databázi pro rekonstrukci.

Kompletní postup rekonstrukce je popsán na Obr 4.4.



Obr 4.4: Schéma postupu při rekonstrukci

5 Implementace

V následující kapitole budou popsány postupy a technologie zvolené pro vytvoření multiplatformní aplikace pro rekonstrukci obličejů.

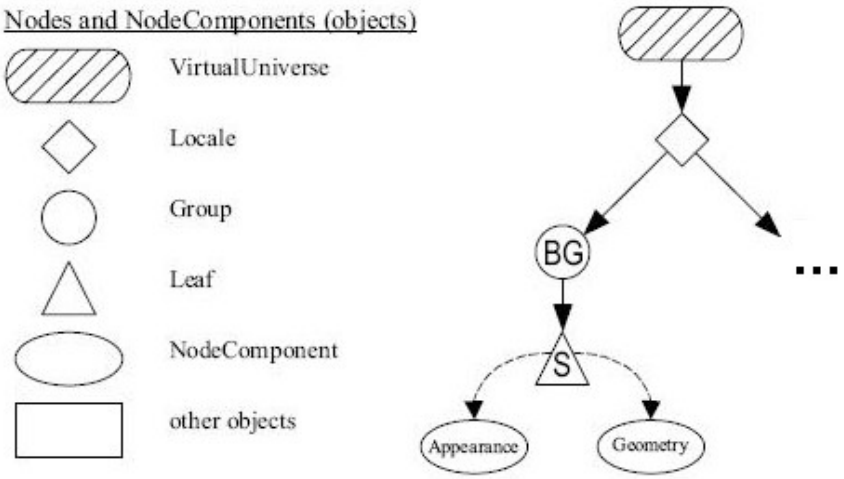
5.1 Jazyk a knihovny

Pro implementaci byl zvolen jazyk Java, se kterým jsem se seznámil během kurzu IJA. Volba byla ovlivněna zejména osobními preferencemi a dostupností manuálů a knihoven. Výhoda Javy spočívá také v přenositelnosti, kdy je kód přeložen pouze do mezikódu a ten je následně interpretovaný virtuálním strojem Java. Pro zobrazování 3D modelu byla zvolena knihovna Java 3D, která je kompatibilní s grafickými komponentami Swing a AWT jazyka Java. Hledání vhodné knihovny pro tvorbu a práci se statistickým tvarovatelným modelem bylo náročnější a probíhalo metodou pokus-omyl. Vyzkoušeny byly knihovny jamal [24] nebo PCA [25]. Neúspěch využití těchto knihoven byl zaviněn zejména neznalostí a kusými návody pro implementaci. Implementace těchto knihoven vedla k neoptimálnímu využití paměti s následkem pádu aplikace. Jako optimální byla zvolena knihovna OpenCV.

5.1.1 Java 3D

Java 3D je API vyvinuté v Sun Microsystems pro renderování interaktivní 3D grafiky využívající jazyka Java. Jedná se o klientské rozhraní s možností využití běžně používaných Swing či AWT prvků. Běh tohoto API spoléhá na OpenGL či DirectX, které provedou renderování scény, zatímco logika a kontrola scény zůstává na programátorovi. OpenGL nabízí vyšší úroveň popisu scény. Hlavní výhodou používání Java3D je programování pouze v jazyce Java, což je pro mnohé programátory velké pozitivum. Programátorům díky vysoké abstrakci nabízí možnosti popisu scény, kde je možné popisovat tvary, materiály, osvětlení a mnohé další. Tyto vlastnosti jsou ukládány do scénového grafu (viz Obr 5.1). Jedná se o vyšší úroveň popisu scény, který umožňuje jednoduše popsat scénu a manipulovat s objekty na ní. Tato vysoká abstrakce však nemusí vyhovovat všem. Zkušení OpenGL programátoři jsou schopni dosáhnout lepších výsledků přímým programováním OpenGL [26].

Vytváření scény se skládá z několika kroků. Je nutné dodržet postup vytváření grafu scény, aby byly zobrazeny všechny požadované objekty.



Obr 5.1: Základní prvky pro vytváření grafu scény v Java3D

Postup při vytváření scény:

1. Vytvoření 3D prostoru (VirtualUniverse)
2. Vytvoření kořenového uzlu (BranchGroup – BG)
3. Vytvoření geometrie objektu a vzhledu objektu (Shape – S)
4. Přidání osvětlení do scény

5.1.2 OpenCV

Open source Computer Vision Library je open source a BSD licencovaná knihovna obsahující několik stovek algoritmů pro počítačové vidění. Je určená pro komerční i akademické použití. Poskytuje rozhraní pro jazyky Java, Python, Matlab, C++ a C na platformách Windows, Linux, Mac OS, iOS či Android. Knihovna je na napsaná v optimalizovaném kódu C/C++ s možnostmi pro vícejádrové zpracovávání. Byla vytvořena pro výpočetní efektivitu se zaměřením na real-time aplikace.

Knihovna obsahuje více než 2500 algoritmů, které je možné využít pro detekci a rozpoznání obličejů, sledování pohybu objektů, práci s 3D modely, sledování pohybu očí apod. Ocenitelnou vlastností knihovny je automatické ovládání paměti zahrnující alokaci, realokaci či dealokaci paměti pro výstupní data. Dalšími vlastnostmi jsou multi-threding, „re-enterability“ (víc vláken může využívat reference) a možnosti odchycení výjimky nebo chyby.

Při implementaci byl využit kompaktní modul Core obsahující základní struktury včetně využité struktury Mat a metody používané v ostatních modulech.

5.2 Tvorba databáze

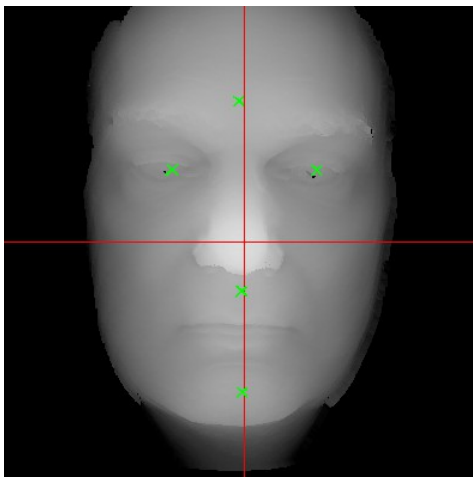
Při tvorbě databáze bylo nutné rozhodnout, jak bude databáze reprezentována. Protože bylo nutné zarovnat modely na sebe tak, aby s nimi bylo možné dále pracovat. Vhodný řešením bylo převést 3D model do hloubkové mapy. Tímto způsobem bylo možné vytvoření trojrozměrného pole, ze kterého se indexací souřadnic X a Y a čísla modelu dá vyčíst kterýkoliv bod jakéhokoliv modelu. Nutné bylo stanovit velikost hloubkové mapy. K tomu došlo pomocí experimentování s velikostí a volbou vzhledem k velikosti monitoru. Rozlišení hloubkových map bylo stanoveno na 400×400 pixelů. Pro manipulaci s databází byla vytvořena třída DB. Databázi je možné vytvářet přímo v programu. Uživateli jsou poskytnuty funkce jako přidávání obličejů do databáze, odstranění obličeje, dodatečná interpolace či načítání a ukládání databáze ve formátu .bin.

5.3 Načtení a zarovnání modelů

Pro všechny modely načítané programem je uplatněný stejný postup. Model je nejdříve načten, proběhne volba bodů s následným automatickým zarovnáním a nakonec uložení do odpovídající třídy ve formě hloubkové mapy. Vzhledem k rozšířenosti formátu .obj není implementace načtení v Javě problematická. Aby bylo docíleno jednotné velikosti, bylo využito příznaku ObjectFile.RESIZE. Dojde tak k přepočítání dat do rozsahu <-1, 1> ve všech dimenzích.

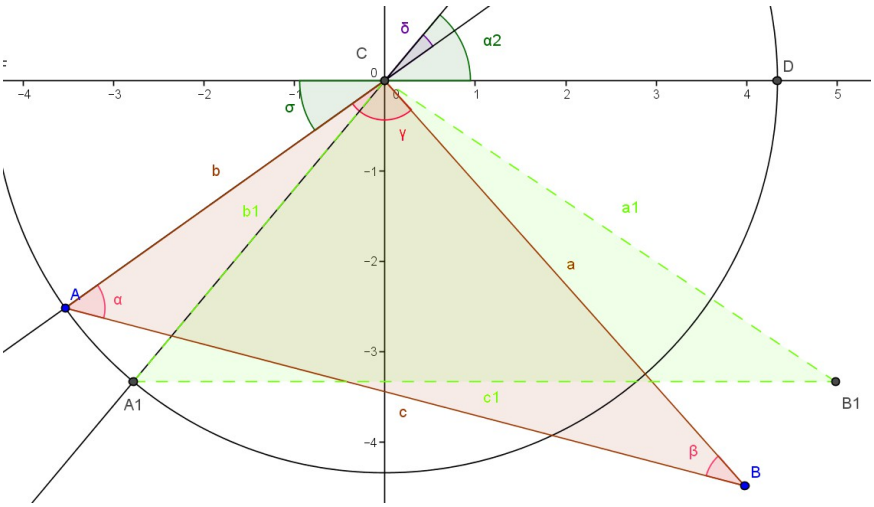
```
ObjectFile objFileloader = new ObjectFile(ObjectFile.RESIZE);
Scene scene = objFileloader.load(filename);
```

Výběr bodů probíhá na hloubkové mapě (Obr 5.2), kde je možné přesněji vybírat body. V případě nutnosti je možné i modelem rotovat ve všech osách. Vzhledem k množství souborů bylo vybírání bodů zrychleno o automatickou změnu na další bod. Zarovnání obličejů bylo rozděleno do 3 kroků, ve kterých postupně proběhne zarovnání podle očí, následně podle nosu a nakonec proběhne vycentrování modelu a případná změna velikosti.



Obr 5.2: Výběr bodů pro zarovnání

Veškeré transformace probíhají se souřadnicemi uloženými do seznamu datového typu Point3f pomocí třídy Tranform3D knihovny Java3D. Díky tomu nedochází k velkým ztrátám přesnosti kvůli zaokrouhlení a výpočet je rychlý. Při zarovnávání modelu probíhají transformace zvětšování, translace a rotace.



Obr 5.3: Rotace trojúhelníku ABC

Pro rotaci je nutné vypočítat úhel δ, o který bude model otočen (viz Obr 5.3). Toho docílíme následujícím výpočtem na příkladu zarovnání očí do stejné hloubky v mapě. Mějme 3 body A (levé oko), B (pravé oko) a C (počátek soustavy souřadnic).

$$A=(x_a, z_a), B=(x_b, z_b), C=(0,0) \tag{23}$$

Pro délky stran trojúhelníku ABC platí vztahy (24).

$$a=\sqrt{(x_a^2+z_a^2)}, b=\sqrt{(x_b^2+z_b^2)}, c=\sqrt{((x_a-x_b)^2+(z_a-z_b)^2)} \tag{24}$$

Pro výpočet úhlu α, který má trojúhelník svírat s osou *X,* aplikuji kosinovou větu a pravidla o velikosti střídavého úhlu.

$$a^2 = b^2 + c^2 + 2 * b * c * \cos\alpha$$
$$\alpha = \arccos\left(\frac{\left(-a^2 + b^2 + c^2\right)}{\left(2 * b * c\right)}\right) \tag{25}$$

Následně vypočítám aktuální úhel σ mezi stranou *b* a osou *x.* Následným odečtem úhlů σ a α získám úhel otočení δ.

$$\sigma = arctng\left(z_a, x_a\right)$$
$$\delta = \sigma - \alpha \tag{26}$$

Výpočet posunutí v osách byl daný požadavky vycentrování očí (translace t_x), posunu očí do určité výšky (translace t_y) a zarovnání do konkrétní hloubky (translace t_z). Zvětšení modelu se vypočítá dle vztahu (28). Konstanty ve vzorcích byly experimentálně zvoleny.

$$t_x = \frac{-\left(x_a + x_b\right)}{\left(2\right)}$$
$$t_y = 0.3 - y_a \tag{27}$$
$$t_z = 0.6 - \frac{\sum_{i=1}^{5} z_i}{5}$$

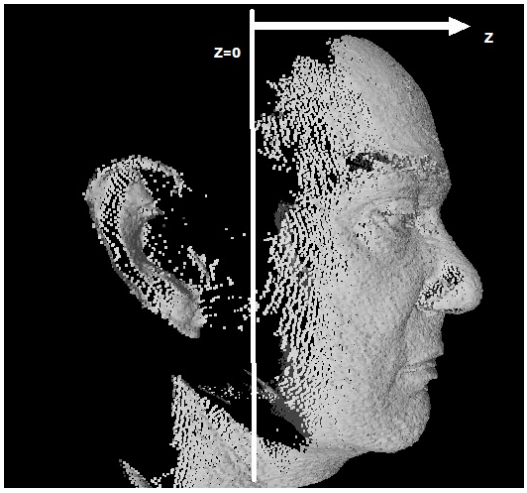
$$s = \frac{0.3_{konst}}{|x_a|} \tag{28}$$

5.4 Hloubková mapa a interpolace

Při tvorbě hloubkové mapy dochází k přepočítání souřadnic modelu na souřadnice do dvourozměrného pole Float [X][Y] = Z, kde *X* a *Y* značí souřadnice a hodnota *Z* je hloubka. Protože přiřazení není jednoduché, je nutné souřadnice pole vypočítávat přímo z hodnot *X* a *Y* každé souřadnice modelu. Je nutné řešit i možné několikanásobné přiřazení do stejného místa. Tento problém byl vyřešen následujícím algoritmem:

```
if (p.x<1 && p.y<1 && p.x>-1 && p.y>-1&& p.z>-1 && p.z<1) {
            getx= (p.x+1)*size/2;
            gety= (p.y+1)*size/2;
            if (RPole[Math.round(getx)][Math.round(gety)]<p.z)
      {
            RPole[Math.round(getx)][Math.round(gety)]=p.z;
      }
}
```

Do *getx* a *gety* přiřazuji hodnoty z intervalu <-1,1> přepočítané do intervalu <0,velikost řádku/sloupce>. Následným zaokrouhlením získám souřadnice hloubkové mapy. Na tyto souřadnice přiřadím hodnotu *Z* je-li větší než původní, přičemž hodnoty jsou inicializované na 0. Dojde tak k ořezání části modelu, který ani nebude modelován (viz Obr 5.4).



Obr 5.4: Ořezání modelu při tvorbě hloubkové mapy

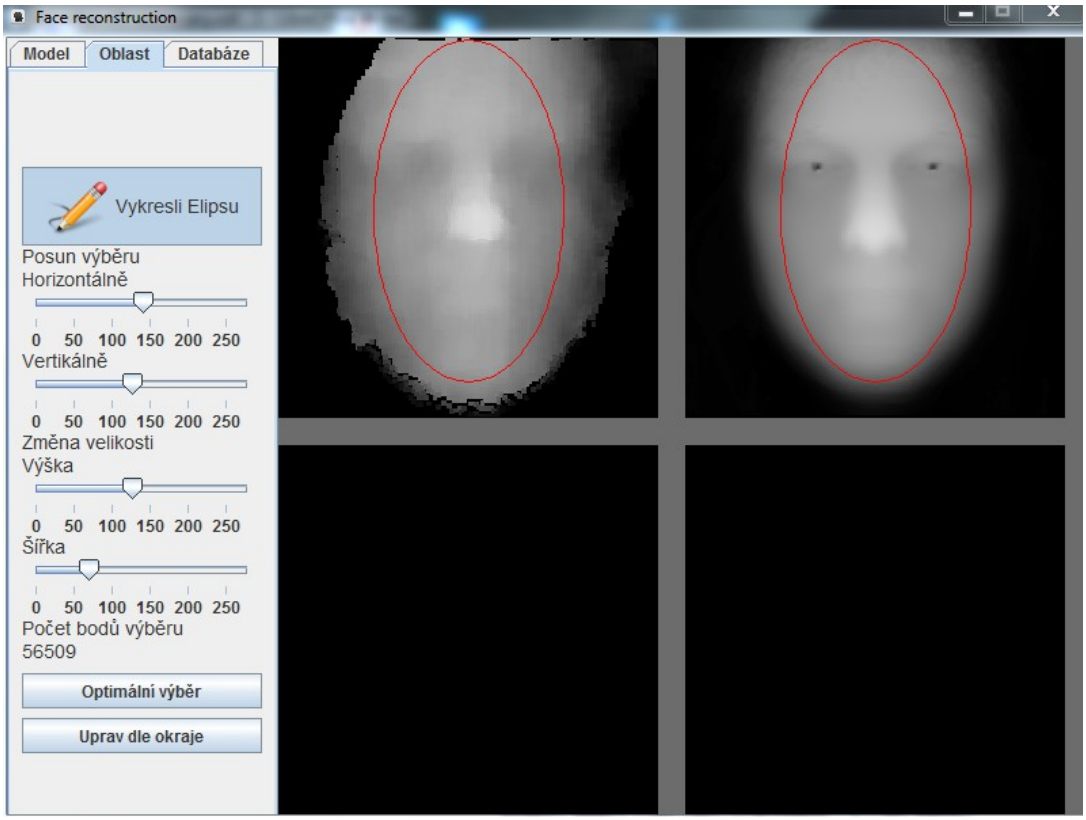
Vzhledem k chybám v hloubkové mapě vzniklým při skenování či zvětšování modelu bylo nutné doplnit chybějící body. Návrh však počítal s manipulací a výběrem bodu přímo z hloubkové mapy. Při snaze vyhnout se problémové manipulaci s načtenými daty byla navržena interpolace pomocí mřížky 3×3. Ta dosazuje na chybějící místa hloubkové mapy průměr z okolních bodů. Aby nedocházelo k vysoké degradaci a velkým změnám, pracuje mřížka pouze s kladnými nenulovými hodnotami. Nedochází tak ke snižování dosazované hodnoty vzhledem k nedostatku okolních bodů. Negativním důsledkem je doplnění bodů i mimo obličej, v závislosti na průchodu 2D polem hloubkové mapy. To ničemu nevadí, protože pro rekonstrukci bude vybrána pouze část obličeje obsahující oči, ústa a nos.

5.5 Výběr plochy pro rekonstrukci

Jak už bylo zmíněno výše, pro rekonstrukci je vybrána jen část obličeje. Je to z toho důvodu, že se na 3D snímcích vyskytují obličeje s vlasy zasahujícími do obličeje, bez vlasů, s krkem či bez, s čepicí apod. Jedná se o vlastnosti, které se nedají věrohodně statisticky namodelovat, a proto budou z procesu rekonstrukce vyjmuty.

Výběr plochy je přenechán uživateli, aby mohl v závislosti na databázi vhodně zvolit plochu pro rekonstrukci (viz Obr 5.5). Je tak možné zvolit pro rekonstrukci pouze část obličeje (např. nos).

Omezující vlastností může být tvar plochy ve tvaru elipsy. Ta byla zvolena jako přirozený tvar obličeje. Pro manipulaci má uživatel k dispozici 4 posuvníky pro zvětšování a posouvání vybrané oblasti a informaci o počtu vybraných bodů. Je možné využít funkcí pro automatický výběr dle databáze nebo kontrolu hranic.



Obr 5.5: Okno výběru oblasti pro analýzu a rekonstrukci

5.6 Analýza a rekonstrukce

K provedení PCA analýzy a promítání modelu do a ze statistického modelu byly využity funkce modulu Core knihovny OpenCV.

5.6.1 Analýza

Vybraná oblast obličeje je pomocí funkce getCircledData_Mat() převedena na sloupcový vektor a následně vložena do matice X typu Mat obsahující všechny modely. Pro vypočítání vektorů pak stačí zavolat funkci Core.PCACompute(), která vypočítá jak průměrný vektor, tak vlastní vektory.

Vzhledem k návrhu počítajícího s ořezáním dat je nutné vypočítat i vlastní čísla. Toho docílíme pomocí funkcí Core.calcCovarMatrix() a Core.eigen().

5.6.2 Rekonstrukce

Rekonstrukce obličeje probíhá v pěti krocích:

- Převedení modelu na vektor (funkce getCircledData_Mat)
- Promítnutí do podprostoru pomocí statistického modelu (funkce Core.PCAProject)
- Oříznutí jednotlivých prvků získaného vektoru
- Promítnutí z podprostoru pomocí statistického modelu do prostoru obličeje (funkce Core.PCABackProject)
- Převedení do hloubkové mapy nebo 3D modelu z vektoru

Pro uživatele je ponechána volba počtu vektorů ve formě posuvníku pro rekonstrukci dat. Pomocí zobrazených informací je možné sledovat, jak se model při různém počtu použitých vektorů liší vůči vzoru před rekonstrukcí.

5.6.3 Porovnání modelů

Po uložení modelu v okně PCA analýzy je možné získaný model porovnávat pomocí 5 zobrazovacích funkcí. Ty umožňují model porovnat například pomocí rozdílových map[3]. K dispozici jsou dva typy map. Základní mapa zobrazí absolutní rozdíl hloubkových map, který je možné zobrazit i v 3D prostoru. Barvená hloubková mapa popisuje velikost rozdílu jednotlivých bodů (viz Tabulka 3).

Tabulka 3: Význam barev

Barva	Rozdíl
Červená	0-0.005
Oranžová	0.005-0.01
Žlutá	0.01-0.05
Zelená	0.05-0.1
Modrá	0.1-1

Další možností porovnání je pomocí rozpůlení modelu na levou a pravou tvář nebo pomocí průhlednosti zobrazení obou modelů přes sebe.

3 Hodnoty hloubkové mapy jsou v rozsahu <0,1>.

6 Testování

Pro otestování programu bylo navrženo několik testů zaměřených na různé části implementace.

1. Rekonstrukce modelu z databáze vyřazeného
2. Rekonstrukce modelu v databázi obsaženého
3. Rekonstrukce modelů pořízených Kinectem
4. Porovnání výsledků při použití různého počtu vlastních vektorů

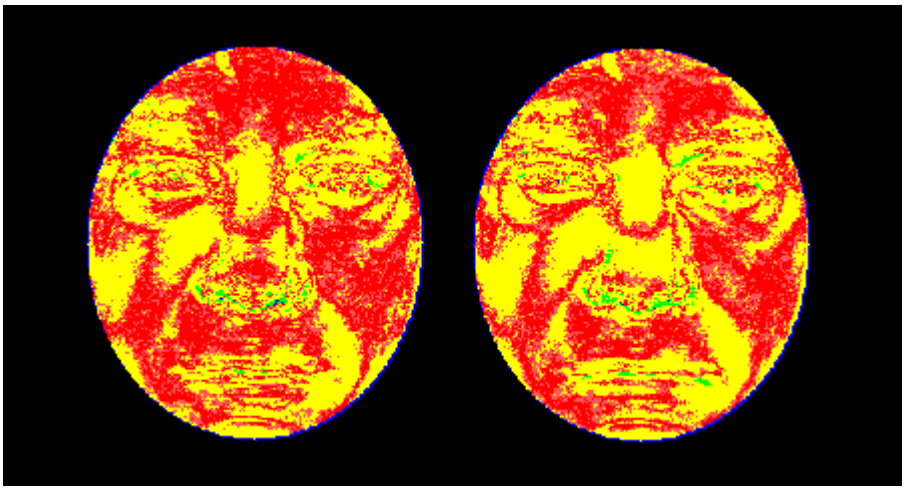
Při testování rekonstrukce byla využita databáze mužů (40 modelů), žen (32 modelů) a společná (72 modelů).

6.1 Model vyřazený z databáze

Test měl za cíl zjistit, jak moc se změní obraz větší kvality při snaze jej zrekonstruovat. Z databáze byl vybrán a smazán obraz, který byl následně předložen jako neznámý model k rekonstrukci.

Obr 6.1: Test 1 – model z databáze vyřazený, rekonstrukce pomocí společné databáze, rekonstrukce pomocí databáze mužů

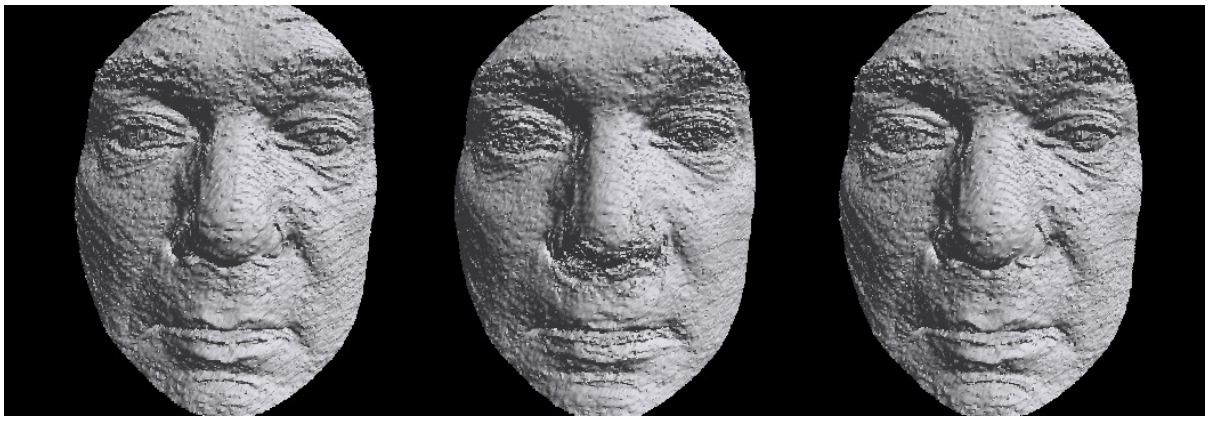
Můžeme vidět, že při snaze o rekonstrukci modelu z databáze vyřazeného, dostáváme obličeje, které se původnímu příliš nepodobají (viz Obr 6.1). Rekonstrukce byla provedena nad 52 761 souřadnicemi při použití všech možných vlastních vektorů (71 – společná databáze, 39 – databáze mužů). Jak je patrné z rozdílových map (viz Obr 6.2), při použití menší databáze dochází k horší rekonstrukci. Průměrná změna při použití velké databáze byla 0,011 16 bodu, zatímco při použití pouze databáze mužů 0,012 6 bodu.



Obr 6.2: Test 1 – rozdílová mapa při použití společné databáze (vlevo) a databáze mužů (vpravo)

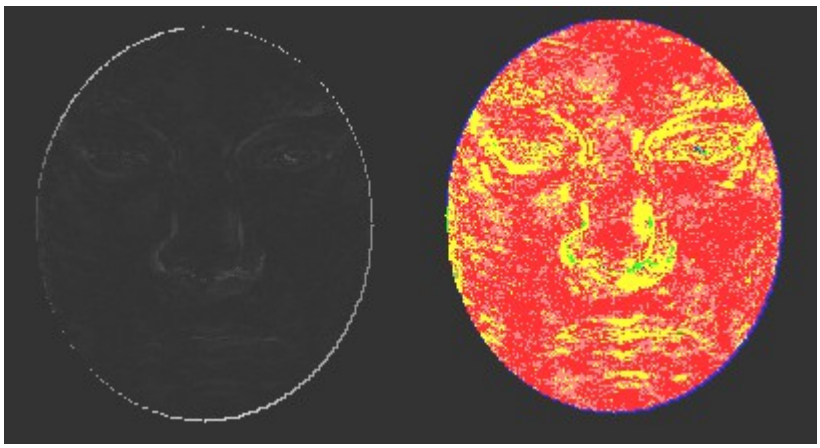
6.2 Modely obsažené v databázi

Cílem testu bylo zjistit funkčnost promítání obrazu do databáze a jeho zpětné rekonstrukce za předpokladu, že testovaný obraz je již v databázi obsažen. Aplikaci byl předložen jako neznámý model pro rekonstrukci.



Obr 6.3: Test 2 - původní model, rekonstruovaný model, model v databázi

Na Obr 6.3 je zobrazen model před a po rekonstrukci ve srovnání s originálem v databázi. Vidíme, že modely před a po rekonstrukci si odpovídají a změny jsou oproti testu 1 (viz 6.1) minimální. Fakt, že je testovaný model již v databázi obsažen, má tedy na výsledek významný vliv. Pro zrekonstruování modelu bylo použito všech 72 vlastních vektorů ze společné databáze (72 osob). Rekonstrukce se týkala 52 761 souřadnic. Pomocí rozdílové mapy Obr 6.4 (původní a rekonstruovaný model) lze vidět změny, ke kterým došlo. Celkový rozdíl obou modelů je 911,710 08 bodů, což činí průměrnou změnu jednoho bodu o 0,017 28 bodu.



Obr 6.4: Test 2 - mapa rozdílu a barevná rozdílová mapa

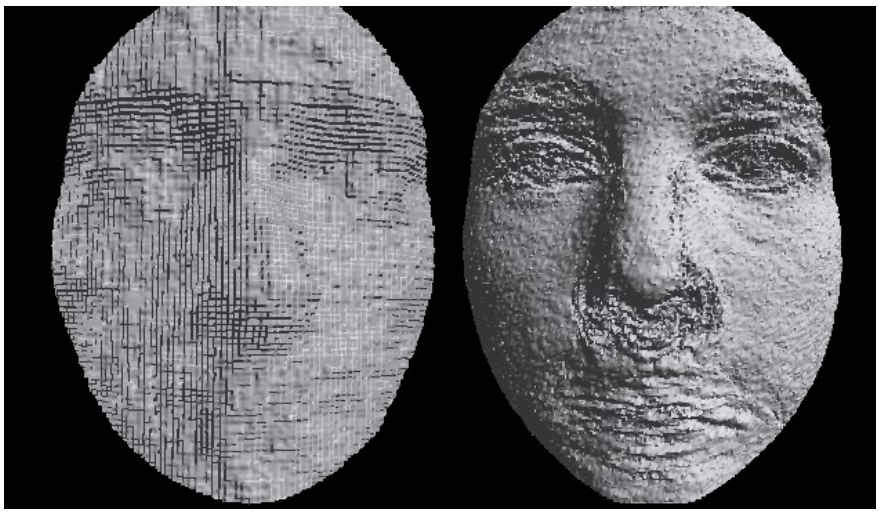
6.3 Modely z Kinectu

Tento test ověřoval dosažení hlavního cíle projektu, a to rekonstrukci modelů pořízených pomocí zařízení Kinect. Nízkou kvalitu modelů pořízených Kinectem můžeme poznat již po načtení modelu. Díky nižší kvalitě se hůře vybírají body pro zarovnání, což může mít za následek horší výsledek rekonstrukce. Rekonstrukce byly provedeny pomocí velké databáze. Oblast pro rekonstrukci byla zvolena pomocí funkce vybírající optimální oblast ze zvolené databáze. Optimální oblast obsahovala, tak jako v předešlých testech, 52 761 souřadnic. Test proběhl na 5 modelech z Kinectu.

Tabulka 4: Vlastnosti rekonstrukcí modelů z Kinectu

Rekonstrukce	Model	Průměrná změna	Celková změna	Počet bodů
1	1-20140123123528.obj	0,010 31	543,736 06	52 761
2	2-20140123123513.obj	0,012 72	672,403 41	52 761
3	5-20140123123752.obj	0,009 44	497,979 55	52 761
4	4-20140123123716.obj	0,020 02	1 056,191 63	52 761
5	3-20140123123626.obj	0,018 33	967,039 22	52 761

V tabulce (viz Tabulka 4) je možné vidět, jak velké změny byly na modelech provedeny. Na obrázku Obr 6.5 a Obr 6.6 jsou zobrazeny nejúspěšnější a nehorší rekonstrukce v závislosti na parametru změny.

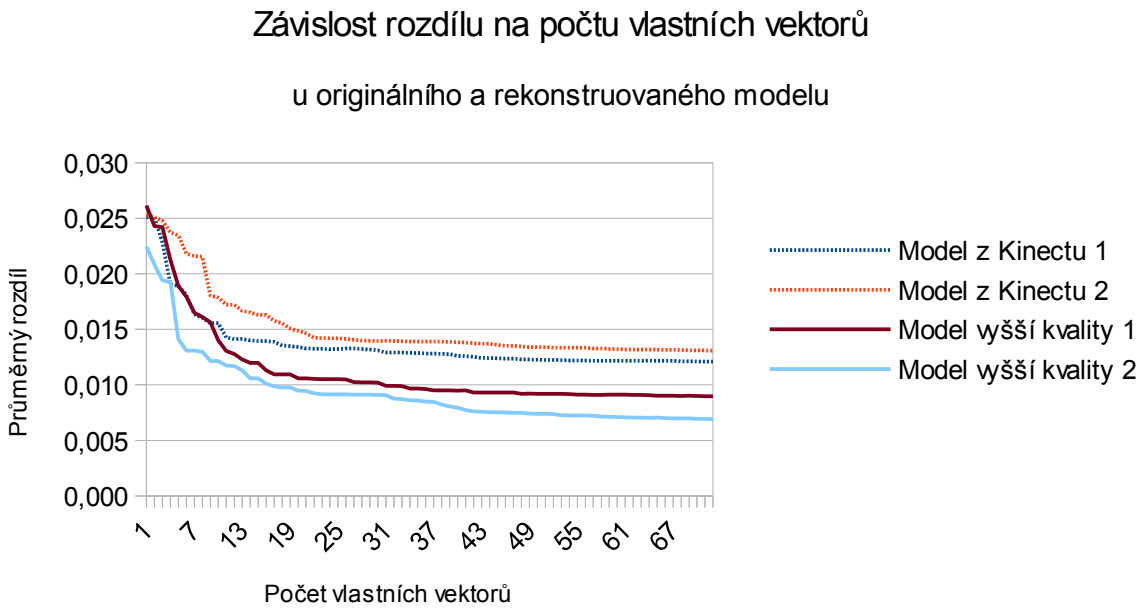


Obr 6.5: Nejlepší rekonstrukce modelu z Kinectu

Obr 6.6: Nejhorší rekonstrukce modelu z Kinectu

6.4 Použití různého počtu vektorů

Cílem tohoto testu bylo zjistit, kolik vektorů je vhodných pro rekonstrukci. Konkrétními parametry testu byla závislost počtu vektorů na průměrné změně modelu a vizuální porovnání. Pro testování byla zvolena společná databáze, z níž byla vybrána oblast o 52 761 souřadnicích. Na této databázi byly otestovány 4 modely (2 modely z Kinectu, 2 kvalitní modely z FRGC). Ty byly postupně promítány a zpětně rekonstruovány pomocí různého počtu vektorů.



Obr 6.7: Graf závislosti rozdílu modelů na počtu vlastních vektorů

Jak je z grafu (viz Obr 6.7) patrné, používání více vektorů při rekonstrukci obličejů se postupně stává neefektivním. Největších změn je dosahováno při využití do 20 vlastních vektorů. Nicméně k dosažení nejlepšího výsledku je ideální využít všech dostupných vektorů.

Obr 6.8: Rekonstruovaný model při použití 1, 5, 10, 20, 40, 72 vlastních vektorů

Při použití většího počtu vlastních vektorů však modely začínají ztrácet na kvalitě, jak jde vidět na Obr 6.8. Chyba se projevuje zejména v oblasti úst a nosu, kde dochází k zobrazení více různých úst a nosů najednou. Vzhledem k dosaženým výsledkům byl ideální počet vektorů u společné databáze stanoven na 20 vlastních vektorů.

7 Závěr

Cílem této práce bylo navrhnout a implementovat řešení, pomocí něhož by bylo možné vylepšit modely obličeje pořízené zařízením Kinect. Výsledkem je multiplatformní aplikace implementována v jazyce Java s využitím knihoven Java3D a OpenCV.

Vytvořená aplikace umožňuje kromě rekonstrukce i prohlížení modelů a porovnávání rekonstruovaných modelů s originálem. Díky využití rozšířeného formátu Wavefront .obj lze program využít i pro rekonstrukce modelů nepocházejících ze zařízení Kinect.

Při testování programu bylo zjištěno, že není vhodné využívat pro rekonstrukci modelu všechny dostupné vektory. Výběrem menšího počtu vektorů dosáhneme hladšího modelu, který sice nemusí odpovídat tváři, ze které model pochází, ale vyhneme se interferenci více modelů nosu či úst. Pokud bychom chtěli rekonstruované modely použít pro rozpoznávání, je naopak vhodné použít všechny dostupné vlastní vektory. Řešením interference kolem úst by pak mohlo být přesnější a robustnější zarovnání jak vstupních modelů pro PCA, tak testovaných modelů.

Tento program je možné dále rozšířit o další funkce vylepšující kvalitu výsledného modelu. Například je možné program rozšířit o barevné spektrum přidáním dalších 3 dimenzí do procesu rekonstrukce. Vhodným nástrojem by mohlo být automatické zarovnání modelů, které by vycházelo z identifikace kontrolních bodů.

Literatura

[1] FOFI, David, Tadeusz SLIWA a Yvon VOISIN. A comparative survey on invisible structured light. In: [online]. 2004, s. 90–98

[2] WILL, Peter M. a Keith S. PENNINGTON. Grid coding: A preprocessing technique for robot and machine vision. *Artificial Intelligence*. 1972, roč. 2, č. 3, s. 319–329

[3] PENG, Tao a Satyandra K. GUPTA. A computational framework for point cloud construction using digital projection patterns. In: *ASME 2006 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference* [online]. B.m.: American Society of Mechanical Engineers, 2006, s. 311–323

[4] *Konica Minolta 3D Laser Scanners | VIVID 9i | VIVID 910* [online]. Dostupné z: http://www.3dscanco.com/products/3d-scanners/3d-laser-scanners/konica-minolta/

[5] *NextEngine 3D Laser Scanner* [online]. Dostupné z: http://www.nextengine.com/products/scanner/specs

[6] ZHANG, Zhengyou. Microsoft kinect sensor and its effect. *MultiMedia, IEEE*. 2012, roč. 19, č. 2, s. 4–10

[7] *Microsoft Word - Kinect Imaging.doc - KinectImaging.pdf* [online].Dostupné z: http://fivedots.coe.psu.ac.th/~ad/jg/nui13/KinectImaging.pdf

[8] OpenNI community developers. *PrimeSense* [online]. Dostupné z: http://www.primesense.com/open-ni/

[9] Asus, PrimeSense Reveals Motion Sensing for PC. *Tom's Hardware* [online]. Dostupné z: http://www.tomshardware.com/news/Motion-Sensing-Kinect-PrimeSense-WAVI-Xtion-Xbox-360,11874.html

[10] PrimeSense. *MIT Technology Review* [online]. Dostupné z: http://www2.technologyreview.com/tr50/primesense/

[11] *Computer Vision Final Project: Kinect* [online]. Dostupné z: http://www.cs.virginia.edu/~jfw3x/cs4501/

[12] *Kinect for Windows Sensor Components and Specifications* [online].Dostupné z: http://msdn.microsoft.com/en-us/library/jj131033.aspx

[13] *Patrick Bouffard - Quadrotor + Kinect Project* [online]. [vid. 3. květen 2014]. Dostupné z: http://hybrid.eecs.berkeley.edu/~bouffard/kinect.html

[14] BLANZ, Volker a Thomas VETTER. A morphable model for the synthesis of 3D faces. In: *Proceedings of the 26th annual conference on Computer graphics and interactive techniques* [online]. B.m.: ACM Press/Addison-Wesley Publishing Co., 1999, s. 187–194

[15] TER HAAR, Frank B. a Remco C. VELTKAMP. 3D face model fitting for recognition. In: *Computer Vision–ECCV 2008* [online]. B.m.: Springer, 2008, s. 652–664

[16] KIM, Kyungnam. Face recognition using principle component analysis. In: *International Conference on Computer Vision and Pattern Recognition* [online]. 1996, s. 586–591

[17] FRITSCH, Lukáš. Metoda PCA a její implementace v jazyce C++. *CVUT v Praze [online]* [online]. 2005. Dostupné z: http://dsp.vscht.cz/konference_matlab/MATLAB07/prispevky/fritsch_l/fritsch_l.pdf

[18] WANG, Zhi-Ming a Jian-Hua TAO. Reconstruction of partially occluded face by fast recursive PCA. In: *Computational Intelligence and Security Workshops, 2007. CISW 2007. International Conference on* [online]. B.m.: IEEE, 2007, s. 304–307

[19] *ReconstructMe | Real Time 3D Scanning Software* [online].Dostupné z: http://reconstructme.net/

[20] *Skanect | OpenNI* [online]. Dostupné z: http://www.openni.org/files/skanect/

[21] *Wavefront .obj file* [online]. 2014. Dostupné z: http://en.wikipedia.org/w/index.php?title=Wavefront_.obj_file&oldid=603064030

[22] PHILLIPS, P. Jonathon, Patrick J. FLYNN, Todd SCRUGGS, Kevin W. BOWYER, Jin CHANG, Kevin HOFFMAN, Joe MARQUES, Jaesik MIN a William WOREK. Overview of the face recognition grand challenge. In: *Computer vision and pattern recognition, 2005. CVPR 2005. IEEE computer society conference on* [online]. B.m.: IEEE, 2005, s. 947–954

[23] COOTES, Timothy F., Gareth J. EDWARDS a Christopher J. TAYLOR. Active appearance models. In: *Computer Vision—ECCV'98* [online]. B.m.: Springer, 1998 [vid. 20. duben 2014], s. 484–498

[24] *JAMA: Java Matrix Package* [online].Dostupné z: http://math.nist.gov/javanumerics/jama/

[25] *Michael Thomas Flanagan's Java Scientific Library: Principal Component Analysis* [online]. Dostupné z: http://www.ee.ucl.ac.uk/~mflanaga/java/PCA.html

[26] SELMAN, Daniel. *Java 3D Programming*. 1st edition. Greenwich: Manning Publications, 2002. ISBN 9781930110359.

[27] PATEL, Ankur a William AP SMITH. 3D morphable face models revisited. In: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on* [online]. B.m.: IEEE, 2009, s. 1327–1334

[28] US DEPARTMENT OF COMMERCE, NIST. *Face Recognition Grand Challenge (FRGC)* [online]. Dostupné z: http://www.nist.gov/itl/iad/ig/frgc.cfm

Příloha A - Obsah CD

Obsah složek přiloženého CD:

- app – obsahuje spustitelnou aplikaci
- doc – vygenerovaná dokumentace
- src – zdrojové kódy aplikace
- bp – text bakalářské práce

Příloha B - Manuál

Požadavky

- OpenGL 1.2 a vyšší
- Java JRE

Windows

Před spuštěním je nutné aktualizovat knihovny .dll v závislosti na 32bitovém či 64bitovém systému. K tomu je možné využít skripty Adjust_x86.bat nebo Adjust_x64.bat. Následně program FaceRec spustíme pomocí skriptu run.bat.

Linux

Nutné instalovat knihovny Java3D. Program je spustitelný pomocí příkazu java -jar FaceRec.jar.

Program

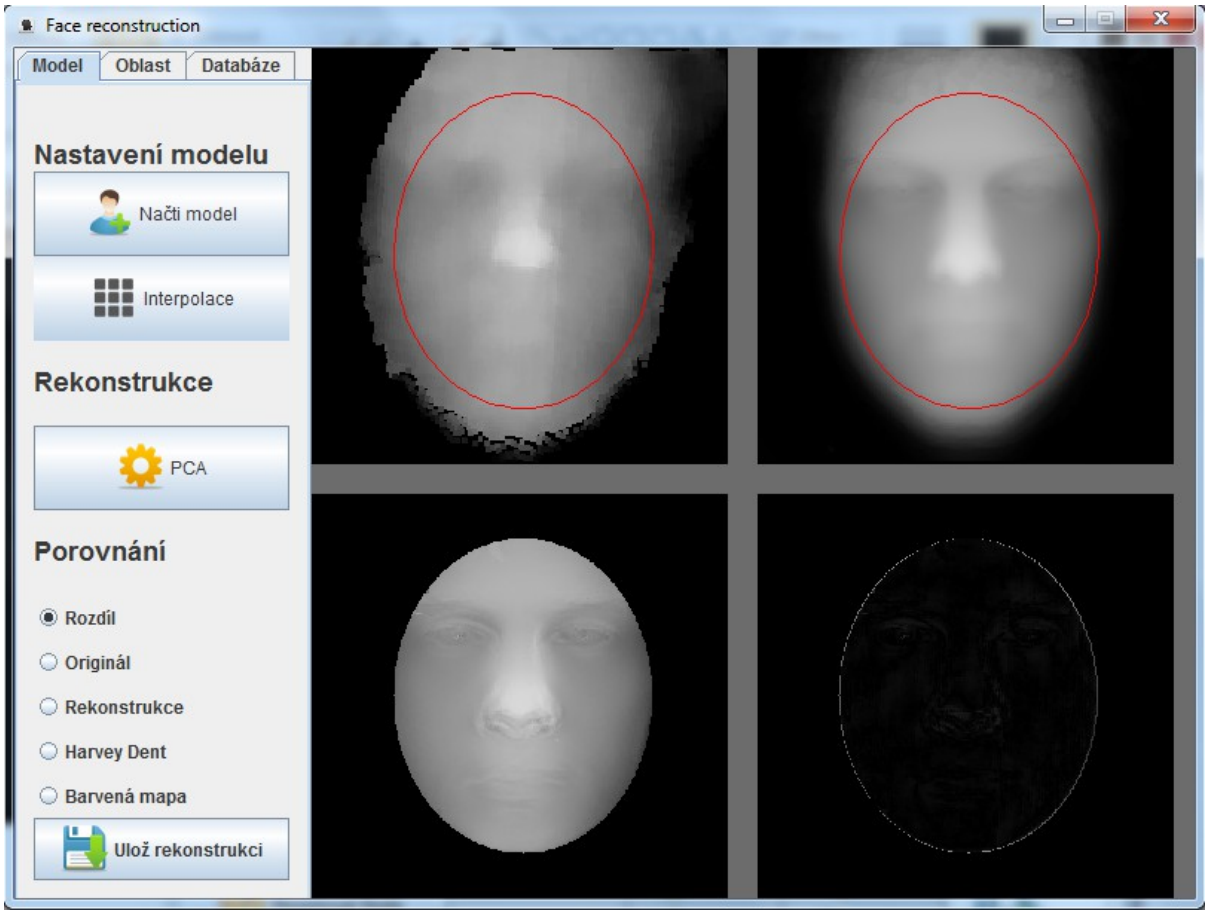
Hlavní okno programu (viz Obr 7.1) obsahuje 3 panely pro ovládání a 4 hloubkové mapy. Ovládací panely (Model, Oblast, Databáze) obsahují nejdůležitější funkce pro ovládání programu. Můžeme zde najít funkce pro načítání modelů, správu databáze, výběr oblasti pro rekonstrukci či spuštění PCA. Výběr oblasti je možný pouze v případě, že je již načtená databáze. Pro výběr jsou implementovány 4 posuvníky pro změnu velikosti a pozice. Výběr je nutné provádět tak, aby elipsa byla uvnitř hloubkové mapy. Pro kontrolu je možné využít tlačítka „Uprav podle okraje“.
Typy hloubkových map hlavního okna:

- Vpravo nahoře – Hloubková mapa databáze
  - Po kliknutí zobrazí 3D model z aktuální hloubkové mapy
  - Procházení modelů pomocí posuvníku v panelu Databáze
  - Možnost zobrazení průměrného obličeje z databáze
- Vlevo nahoře – Hloubková mapa modelu pro rekonstrukci
  - Po kliknutí zobrazí 3D model z aktuální hloubkové mapy
- Vlevo dole – Hloubková mapa rekonstruovaného obličeje
  - Po kliknutí zobrazí 3D model z aktuální hloubkové mapy
- Vpravo dole – různé hloubkové mapy pro porovnávání modelů

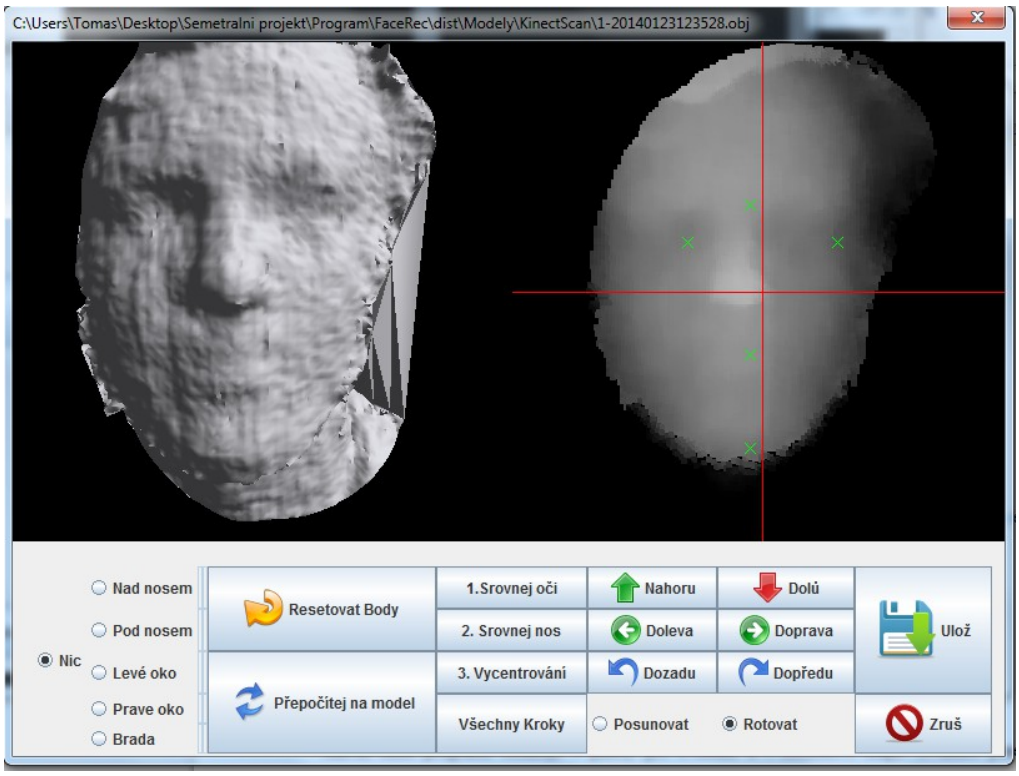
- Po kliknutí zobrazí 3D model z aktuální hloubkové mapy
- Výběr z 5 možností pro porovnání originálu a rekonstrukce

Po zvolení modelu, který se bude přidávat do databáze nebo je určený k rekonstrukci, se otevře okno pro zarovnání modelu (viz Obr 7.2). Zarovnávání probíhá na hloubkové mapě. Pro vybrání bodů je nutné vybrat jednu z možností vlevo dole (Pod nosem, Nad nosem, Levé oko, Pravé oko, Brada). Po každém kliknutí dojde ke změně volby. Po vybrání všech bodů je nutné stisknout tlačítko „Přepočítej na model“. Tím dojde k „přilepení“ bodů na model. Následně je možné začít zarovnávat model pomocí tlačítek „Srovnej oči“, „Srovnej nos“, „Vycentruj“ nebo „Všechny kroky“. Zarovnání se zakončí tlačítkem „Ulož“.

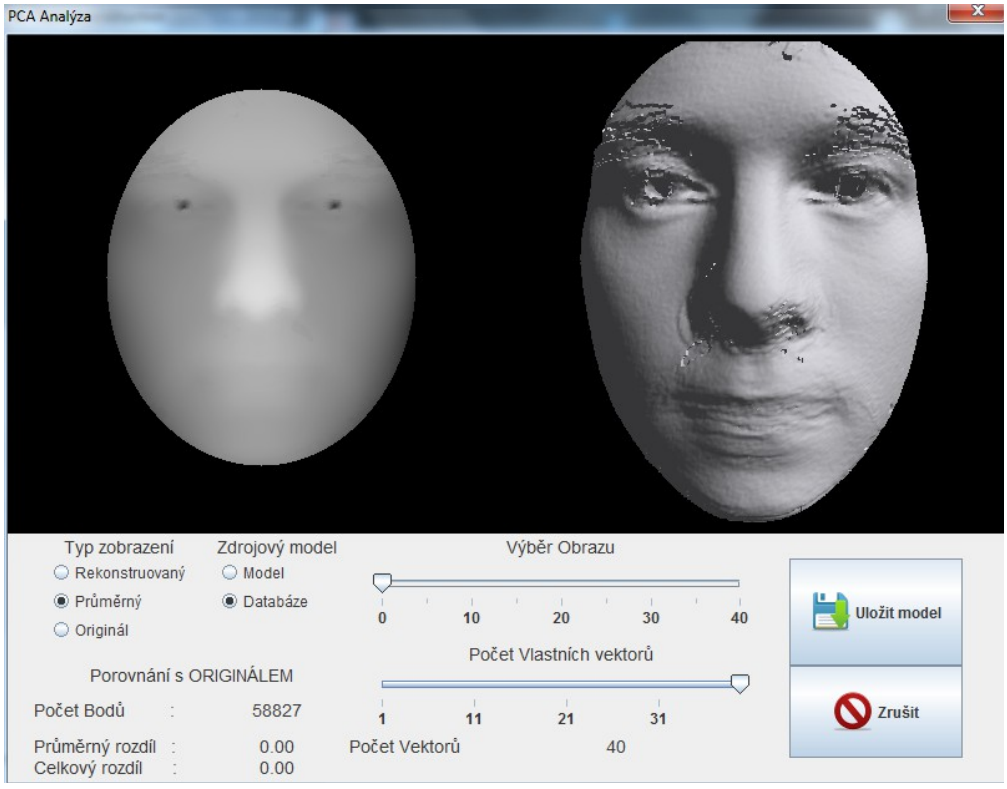
Okno PCA analýzy (viz Obr 7.3) slouží pouze pro procházení modelů a rekonstrukcí. Pomocí posuvníků je možné procházet zrekonstruované modely v databázi či vybírat počet vlastních vektorů pro rekonstrukci. Na hloubkové mapě a 3D modelu lze pozorovat změny. Vybranou rekonstrukci uložíme tlačítkem „Uložit“.



Obr 7.1: Hlavní okno programu



Obr 7.2: Okno načítání modelu



Obr 7.3: Okno pro procházení rekonstrukce